# Snimanje i reprodukcija video signala uz podršku za više kamera zasnovano na FPGA

**Vido, Ivan**



**SVEUČILIŠTE JOSIPA JURJA STROSSMAYERA U OSIJEKU**
**FAKULTET ELEKTROTEHNIKE RAČUNARSTVA I INFORMACIJSKIH TEHNOLOGIJA**

**Sveučilišni studij Mrežne komunikacije**

# Snimanje i reprodukcija video signala uz podršku za više kamera zasnovano na FPGA

**Diplomski rad**

**Ivan Vido**

**Osijek, 2018.**

# Sadržaj

# 1 Uvod

ADAS (engl. *Advanced Driver Assistance System*) su sustavi za pomoć vozačima prilikom vožnje. Većina nesreća događa se zbog ljudske pogreške stoga se ADAS sustavima pokušava umanjiti mogućnost greške, tako što se vozača upozori o stanju na cesti, također pretječe li ga netko, omogući mu se bolja vidljivost po noći koristeći infracrvene kamere te mnoge druge pomoći. Sve više automobilskih tvrtki ulaže u razvoj ADAS uređaja i algoritama. Najznačajniji su Tesla, BMW, Texas Instruments, dok su za našu regiju bitni Rimac Automobili i RT-RK Institut. Testiranje i verifikacija algoritama za pomoć vozaču predstavlja složen proces i zahtijeva vrijeme i resurse ukoliko se izvodi u stvarnom okruženju. Kako bi se testiranje i verifikacija ubrzali, okruženje se snima (video i drugi podaci) i kasnije reproducira u laboratorijima. U tu svrhu razvijaju se uređaji za snimanje i reprodukciju video sadržaja, na primjer AMV *Grabber*.

Zadatak ovog diplomskog rada je napraviti FPGA (engl. *Field Programmable Gate Array*) dizajn koji omogućava AMV *Grabber* ploči primanje i zapisivanje podataka s kamera, te slanje već snimljenog video sadržaja na izlazne portove. Uređaj sadrži centralnu procesorsku jedinicu Zynq XC7Z030. U sklopu zadatka potrebno je realizirati podršku za dva tipa kamera (koje daju RAW i YUV format) koja će video signal s kamera slati putem PCIe sabirnice za potrebe snimanja, odnosno primati video signal putem PCIe i slati na video izlaze za potrebe reprodukcije. Također je potrebno osigurati rad u realnom vremenu (30 slika u sekundi) za do 9 kamera, koje pripadaju podržanim tipovima kamera. Rješenje je potrebno realizirati korištenjem VHDL programskog jezika.

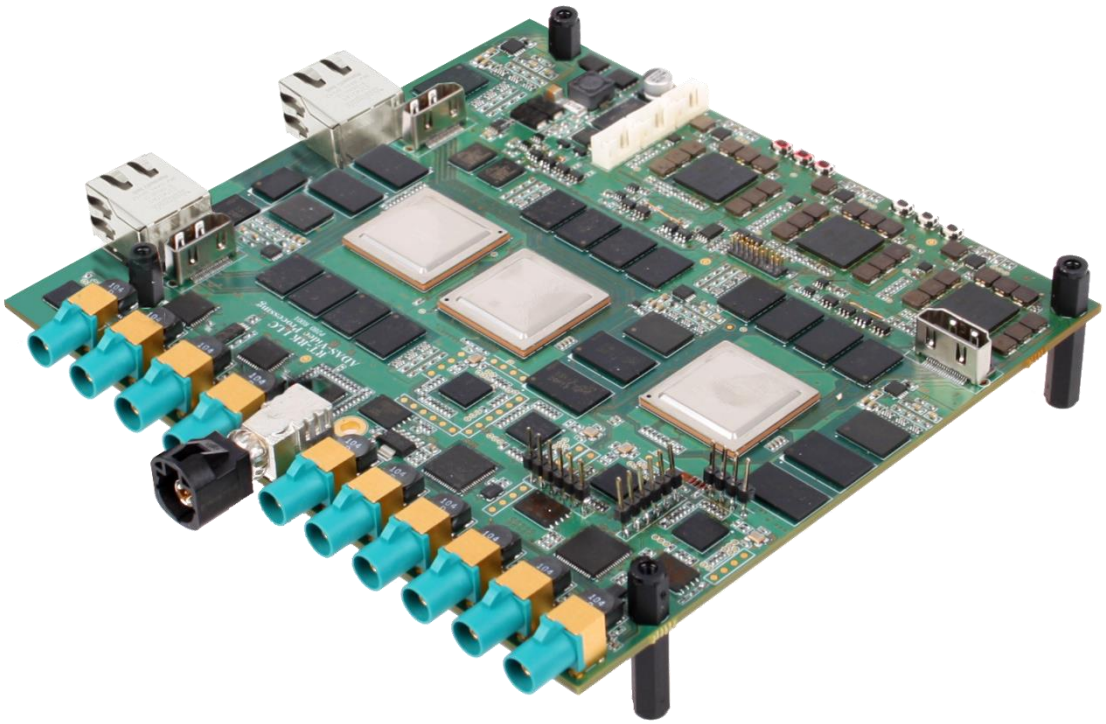
Nadalje, u sljedećem poglavlju nalazi se detaljan opis AMV *Grabber* uređaja i komponenti koje ga čine. U tom poglavlju je opisan FPGA i korišteni blokovi koji su doveli do rješenja. U trećem poglavlju je problem razjašnjen i navedeni su željeni ishodi. U poglavlju 4. Rješenje Problema prikazan je korišteni dizajn i objašnjenja koja omogućavaju repliciranje i verifikaciju rješenja. U tom poglavlju spomenuta su uočena ograničenja dizajna i hardvera AMV *Grabber* uređaja. Posljednje poglavlje tiče se vlastitog verificiranja rješenja uz priložene dokaze.

## 1.1 Pregled postojećih rješenja

Trenutno je tržište ADAS sustava u kontekstu gotovih proizvoda veoma pusto. Postoje brojni algoritmi i specijalizirane komponente poput kamera izvedene specifično za ADAS sustave, ali je ponuda gotovih proizvoda vrlo slaba. Dva takva proizvoda su navedena u nastavku.

**RT-RK AMV Alpha**

Napravljena u suradnji RT-RK Instituta i Texas Instruments, Alpha je prototipna ploča, potpune funkcionalnosti s mogućnosti nadogradnje. Podržava osnovne i napredne sustave upozoravanja vozača, i djelomično autonomne funkcije. Ploča ima deset portova za kamere, koje su zamišljene da pokrivaju sve strane auta, 360º pogled, nadgledaju stanje vozača i imaju mogućnost snimanja po noći. Hardver se sastoji od tri TDA2x SoC-a od kojih je svaki spojen na vlastiti HDMI izlaz, te ima DCAN, Ethernet, JTAG, UART sučelja. Također se može povezati s drugim pločama.



*Sl. 1.1 RT-RK AMV Alpha*

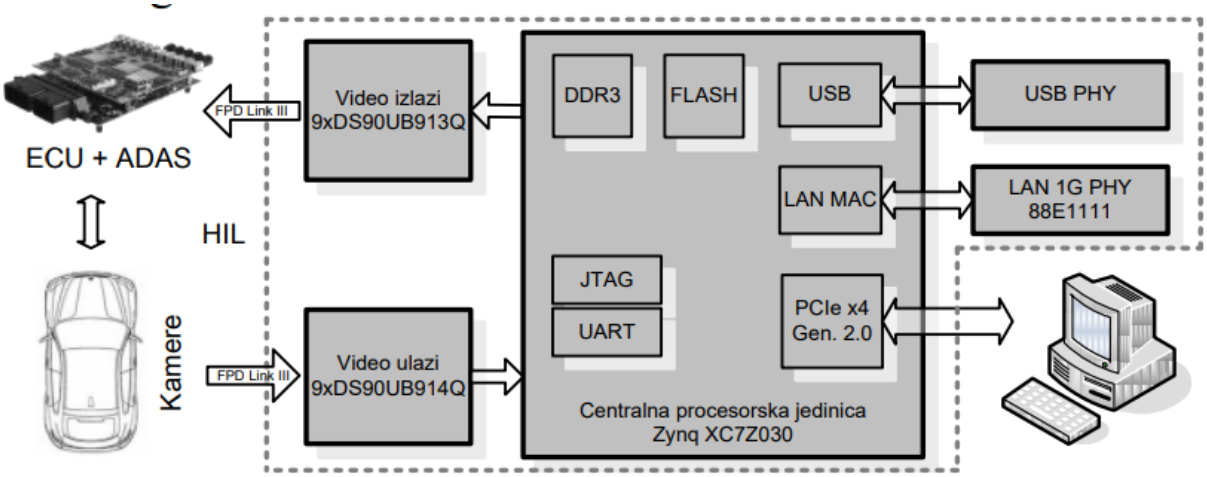
**ARCCORE Arctic Fusion**

Švedska tvrtka Arccore u suradnji s OEM razvili su ADAS prototipnu ploču za razvoj i testiranje. Arctic Fusion je dizajniran u svrhu testiranja tehnologija i koncepata za sigurnost pri vožnji. Uzimajući u obzir naglasak na modularnosti, Arctic Fusion je napravljen s pretpostavkom da će u skoroj budućnosti biti potrebno nadograđivati procesorske sposobnosti. Osnova ploče je Infineon Aurix višejezgreni procesor, a periferna sučelja podržavaju Ethernet, LIN, CAN, FlexRay kanale. Omogućava osam analognih i digitalnih ulaza, uz četiri digitalna izlaza.


*Sl. 1.2 ARCCORE Arctic Fusion,*

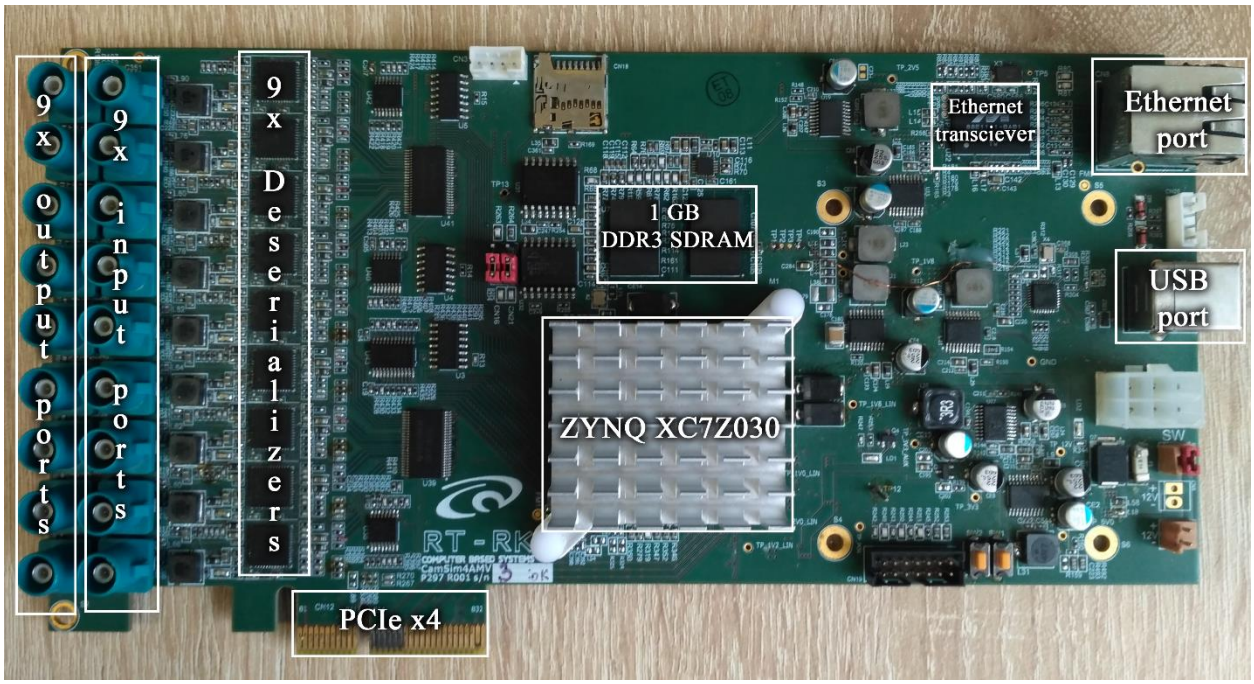
# 2 AMV Grabber

*Automotive Machine Vision Grabber* u punom nazivu, je hardversko okruženje odnosno ploča na kojoj je izvršen ovaj rad. AMV *Grabber* služi za snimanje koristeći više kamera, spremanje snimljenog sadržaja na tvrdi disk i reprodukciju već snimljenog sadržaja u laboratoriju na druge ADAS (engl. *Advanced Driver Assistance Systems*) uređaje. Nadalje AMV *Grabber* uređaj olakšava testiranja ADAS algoritama jer odvaja prikupljanje podataka od korištenja podataka u stvarnom vremenu. Takvim pristupom smanjuju se troškovi ispitivanja i rizici koji nastaju ponavljanjem testova jer se odvijaju u laboratoriju.




*Sl. 2.1 Topologija AMV Grabber uređaja. [2]*

Slika Sl. 2.1 prikazuje arhitekturu AMV *Grabber* uređaja. Na ulazima se nalaze kamere spojene koaksijalnim vodičima (FPD Link III) na *deserializer*-e, na slici navedeni kao „Video ulazi“. Osnova cijelog uređaja je centralna procesorska jedinica Zynq XC7Z030 koja sadrži ARM *Cortex* i FPGA koja je detaljnije opisana u sljedećem potpoglavlju. Komunikacija s uređajem odvija se preko UART sučelja, a programiranje i pokretanje preko JTAG-a. Prijenos podataka između računala i uređaja omogućen je preko tri različita sučelja: USB, Ethernet i PCIe. Radna memorija je 1GB DDR3. Na *flash* memoriju učitava se *boot image* pomoću kojeg se zamjenjuje programiranje i pokretanje uređaja preko JTAG-a.




*Sl. 2.2 Prototip AMV Grabber ploče*

Prototip AMV *Grabber* ploče prikazan je na slici Sl.2.2. Ploču je izradio RT-RK Institut, te je ona unapređenje dosadašnje RT-AV4k.

## 2.1 FPGA

FPGA je kratica od *Field Programable Gate Array* te predstavlja dio procesorske arhitekture zvanom „Sustav na čipu“ odnosno *SoC*-a(*System on Chip*). Programibilnost logičkih sklopova uvelike olakšava prilagođavanje navedenih opće namjenskih *SoC*-eva potrebama proizvođača. Dokaz o višestrukoj namjeni *SoC*-a može se pronaći u granama ugradbenog računarstva kao što su automobilsko računarstvo, u koje pripada i projekt AMV *Grabber*; digitalna televizija, gdje u *SetTopBox*-ovima jezgru također čine *SoC*-evi; medicinska dijagnostika i snimanje; ili bilo kakva zahtjevnija primjena računskih obrada podataka kojoj je neophodna stvarno vremenska i pouzdana obrada podataka. Iz razloga što FPGA jest skupina logičkih sklopova koji svojim rasporedom imitiraju željenu opisanu logičku funkcionalnost, povećava se brzina obrade signala i sklanja teret s osnovnih procesorskih jedinica (CPU).

Korišten *SoC* u AMV *Grabber* projektu je Zynq XC7Z030. Arhitektura pripada u Zynq 7000 familiju koju odlikuje spoj procesorskog sustava (PS) temeljen na ARM *Cortex* procesorima, te programibilne logike (PL). Temeljen na 28nm *high-k metal gate* (HKMG) procesnoj tehnologiji s visokim performansama i niskim utroškom energije. Također, PS

posjeduje vlastitu memoriju i eksterna memorijska sučelja te širok raspon ulazno izlaznih pinova. Procesori u PS-u pokreću se prvi, te tako dopuštaju softverski orijentiran princip konfiguriranja i pokretanja PL-a [3].

Iako je za potrebe ovog rada najbitniji PL dio, u nastavku su navedene komponente koje su korištene radi adekvatnog funkcioniranja same ploče.

UART – engl. *Universal Asynchronous Receiver-Transmitter*, asinkroni *full-duplex* odašiljač i prijemnik s razdvojenim kanalima

PCIe (engl. *Peripheral Component Interconnect Express*) – serijska sabirnica visoke brzine

DDR – engl. *Double Data Rate* sučelje koje podržava DDR2, DDR3, DDR3L i LPDDR2 uređaje

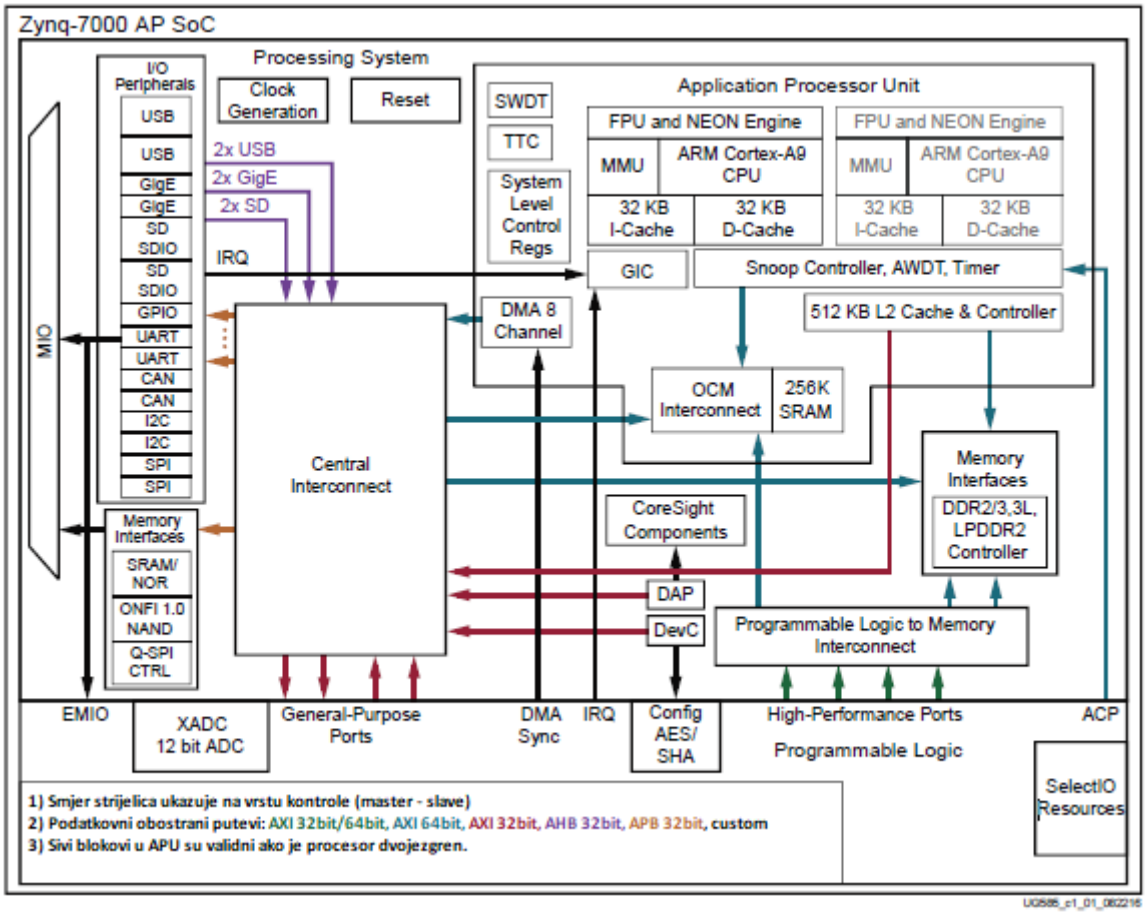
I2C – engl. *Inter-Integrated Circuit*, sinkrona komunikacija između komponenti na ploči, može se primijeniti kao *master* ili *slave* u *multi-master* dizajnu

GPIO – engl. *General Purpose Input-Output*, višenamjenski ulazno izlazni pinovi, raspoređeni u četiri grupe

DMA – engl. *Direct Memory Access*, omogućava prijenos velikih količina podataka bez sudjelovanja procesora

INTERCONNECT – komponenta u PS dijelu SoC-a, uloga joj je povezivanje sistemskih resursa koristeći AXI *point-to-point* kanal. Pristupa joj se preko bloka *AXI Interconnect* u *Vivado®*

INTERRUPTS – upravljač prekida koji dolaze iz programibilne logike ili GPIO pinova.




*Sl. 2.3. Blok dijagram Zynq 7000 [3]*

Na slici Sl. 2.3 mogu se vidjeti relacije navedenih komponenti. Gornji dio slike je PS, koji se u dizajnu koristi kao *ZYNQ7 Processing System*, donji dio je PL koji nema detalja, njega treba dizajnirati koristeći *Vivado*®.

*Vivado*® je okruženje za dizajniranje logičkih sustava baziran na VHDL-u i Verilog-u, proizvođač je Xilinx®. Okruženje pruža korisniku korištenje Xilinx®-ovog intelektualnog vlasništva u obliku već gotovih funkcionalnih blokova i podatkovnih sučelja. Takvi elementi nazivaju se IP (engl. *Intellectual property*).

VHDL je jezik za opisivanje digitalnih elektroničkih sustava. Proizlazi iz programa Ministarstva Obrane Sjedinjenih Američkih Država pod nazivom VHSIC (engl. *Very High Speed Integrated Circuits*) kao potreba za standardizacijom opisivanja struktura i funkcija integriranih krugova. Tako da je pun naziv VHDL-a, *Very High Speed Integrated Circuits Hardware Description Language* [1]. Ispunjava sljedeće potrebe u procesu dizajniranja:

1. Dozvoljava opisivanje strukture sustava, rastavljanjem na podsustave i opisivanjem povezanosti tih podsustava,
2. Omogućava specificiranje funkcionalnosti sistema koristeći poznate oblike programskih jezika.
3. Dozvoljava simulaciju dizajniranog sustava prije nego se proizvede, izbjegavajući tako potrebu za brojnim neuspješnim hardverskim prototipovima.
4. Iz takvog apstraktnog programskog oblika omogućena je sinteza u logički oblik. Time se dizajneri fokusiraju na funkcionalnosti, a ne na sam izgled hardvera.

## 2.2 Podrška za kamere

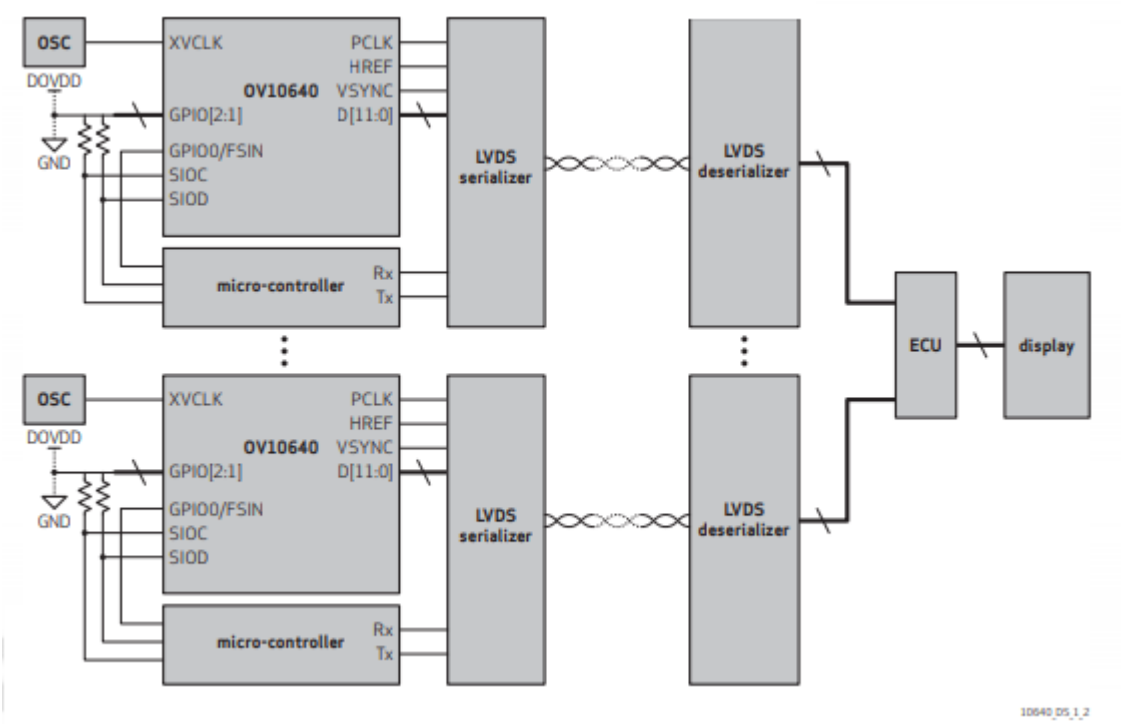
Kao što je u opisu zadatka već navedeno, ploča treba biti u mogućnosti koristiti dva tipa kamera. Kamere koje trebaju biti podržane imaju sljedeće senzore:

1) OmniVision® OV10640
2) OmniVision® OV10635

Glavne razlike između ovih dviju kamera je u formatu slike, gdje kamera 1) podržava samo RAW format s BAYER rasporedom piksela, maksimalne rezolucije 1280x1080, i brojem podatkovnih pinova 12 [4], dok kamera 2) podržava uz RAW format i YUV422 format, maksimalne rezolucije 1280x720, i brojem podatkovnih pinova 10 [5].

RAW format je nekomprimirani, izvorni format kojeg kamera šalje. Dok BAYER raspored piskela označava RGB piksele raspodijeljene po redovima, takvima da je svaki neparni red „RGRG…“, a parni red „GBGB…“ te se time prenosi najviše informacija o zelenoj boji na koju je ljudsko oko najosjetljivije. Zapravo je veoma slično YUV422 formatu, koji prenosi najveću količinu podataka o svjetlini, jer ljudsko oko je osjetljivije na svjetlost nego na boje zbog većeg broja štapića od čunjića. Stoga je raspored YUV piksela bilo koja kombinacija dva „Y“, jednog „U“ i jednog „V“, npr. YUYV.

Ostale funkcionalnosti i specifikacije kamera su gotovo iste, od kojih je najbitnija *serial camera control bus* (SCCB) koji korisniku omogućava gotovo apsolutnu mogućnost prilagođavanja načina rada kamere svojim potrebama, tako što SCCB ima pristup unutarnjim kontrolnim registrima kamere.




*Sl. 2.4 Blok dijagram za primjenu više kamera. Jednaka je primjena i za senzor OV10635 [4].*

Iz slike Sl. 2.4 je vidljivo da se senzor nadovezuje na mikroupravljač i *serializer* koji su također dio kamere. Stoga, kako bi se primali podaci s kamere potrebno je na toj putanji primiti serijske podatke i vratiti ih u paralelni oblik kojeg se može dalje obrađivati. Iz tog razloga se na ploču ugrađuju *deserializer-i* koji čine most između jedne kamere i GPIO pinova na PL komponenti ploče.

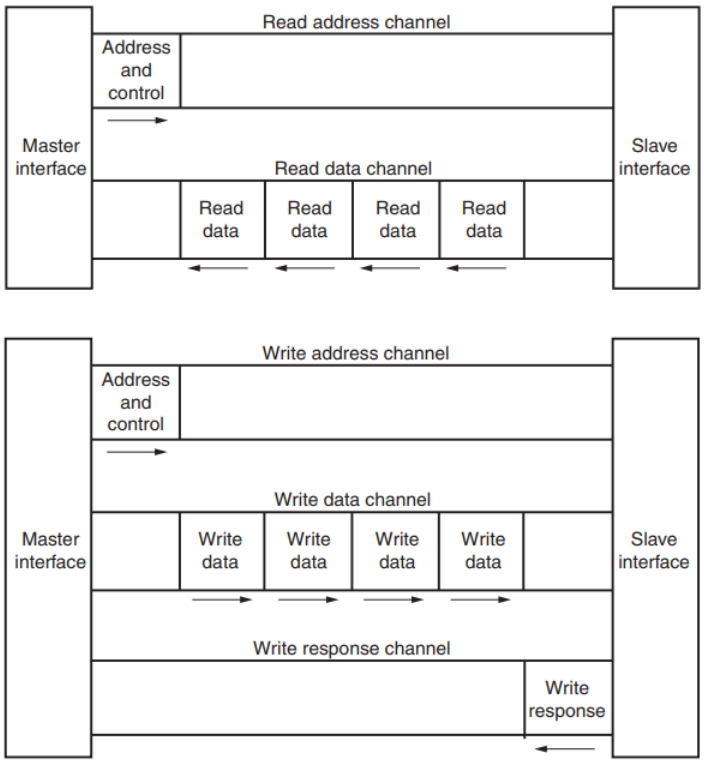
*Deserializer* je tipa DS90UB914A, ulazni serijski signal transformira se u 12-bitni paralelni signal popraćen vertikalnom i horizontalnom sinkronizacijom [6]. Za komunikaciju između *serializer*-a kamere i *deserializer*-a ploče koristi se koaksijalni vodič. Konfiguracija *deserializer*-a vrši se preko I2C protokola, te nakon što je on uspješno konfiguriran preko njega se također preko I2C protokola dolazi do kamere kako bi se i ona konfigurirala za vlastite potrebe.

U paru s *deserializer*-om koristi se i *serializer* tipa DS90UB913A. *Serializer* se koristi za za putanju reprodukcije jer je potrebno paralelne podatke iz memorije pretvoriti u serijske. Konfiguracija se također odrađuje preko I2C protokola. [6].

## 2.3 FPGA blokovi

U ovom potpoglavlju su opisani blokovi koji se koriste u smjeru reprodukcije s ploče. Sučelje za komunikaciju između blokova jest *AXI4-Stream*. To je sučelje koje spaja jedan AXI *master* i jedan AXI *slave* blok, podaci se mogu istovremeno kretati  u oba smjera uz varijabilne

veličine transfera. Na slici Sl. 2.5 vidi se da za potrebe pisanja i čitanja AXI protokol koristi odvojene adresne i podatkovne kanale.




*Sl. 2.5 Prikaz komunikacije u oba smjera [7].*

Protok podataka započinje *master* blokom koji određuje adresu i koju funkciju treba izvesti. Nakon toga, ako je slave blok spreman odraditi zadanu funkciju, započinje „nalet“ (engl. *burst*) podataka, maksimalne veličine 256 segmenata. Iz slike se vidi da pisanje i čitanje funkcioniraju slično, začetnik je *master* blok, jedino u slučaju pisanja potrebno je vratiti potvrdu o uspješnosti zapisanih podataka.

### 2.3.1 *Video In to AXI4 Stream (Video In)*

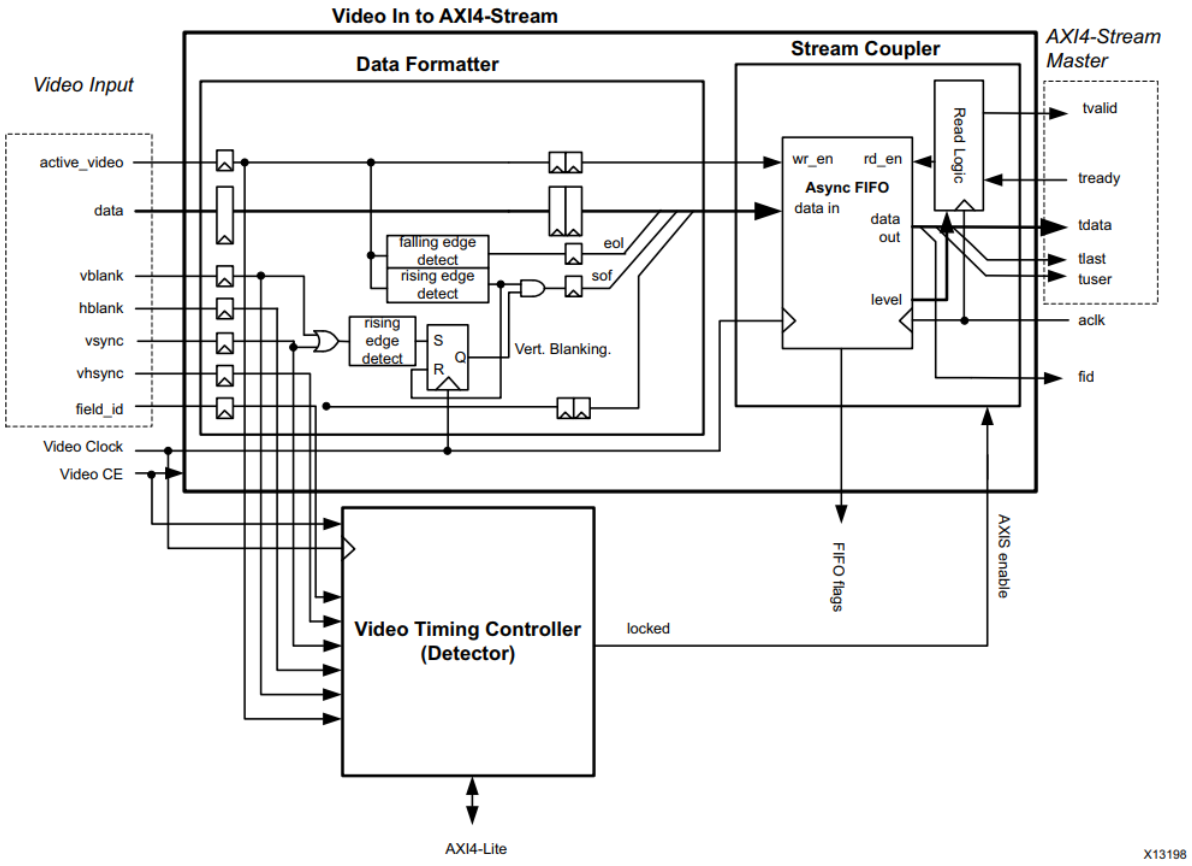
Mnogi blokovi u Vivadu® za obradu videa koriste AXI4-Stream video protokol u svrhu prijenosa videa među blokovima. U većini slučajeva, video se prenosi s eksplicitnim signalima horizontalne (engl. *Hsynch)* i vertikalne sinkronizacije (engl. *Vsynch*), zacrnjivanja (engl. *Blanking*), te valjanih podataka (engl. *Data Valid*). *Video In* blok konvertira ulazni video sa signalima sinkronizacije u *AXI4-Sream* protokol kojim inače *Vivado*® blokovi koriste za obradu video podataka.

*Video In* blok prihvaća ulazni video kao paralelne podatke zajedno s taktom piksela i jednom od ovih kombinacija sinkronizacijskih signala [8]:

- *Vsync, Hsync, Data Valid*
- *Vblank, Hblank, Data Valid*
- *Vsync, Hsync, Vblank, Hblank, Data Valid*

Svaka od ovih kombinacija je zadovoljavajuća za pravilan rad, odnosno za *Video In* nije nužno da su svi signali pristigli na ulaz.

Nadalje, kada je uvjet zadovoljen da je jedna od ovih kombinacija pristigla na *Video In* blok, koristeći unutarnji spremnik, zadane parametre video formata i dobivenu kombinaciju sinkronizacijskih signala *Video In* sliku pretvara u podatke na *AXI4-Stream* sučelju.



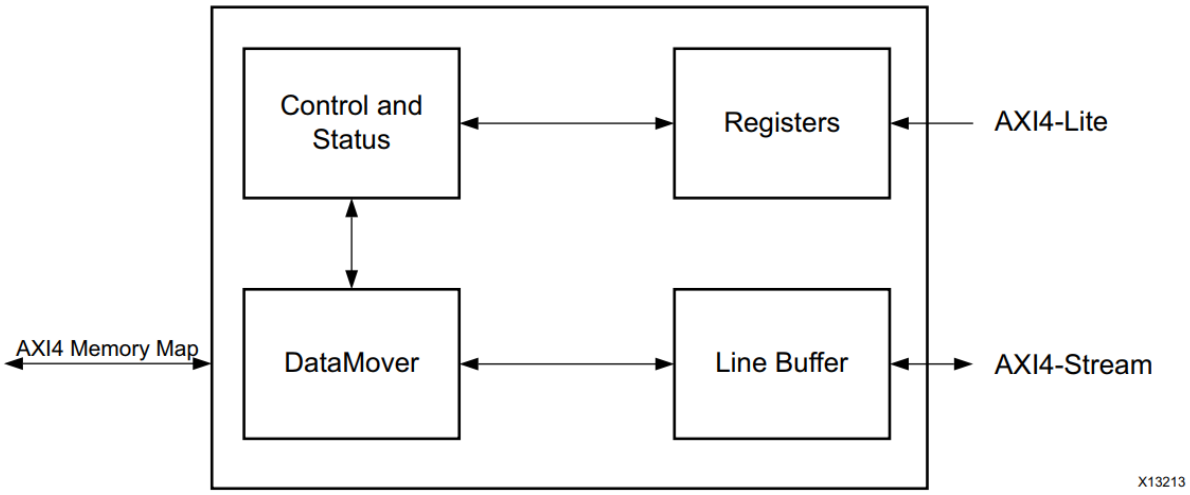

*Sl. 2.6 Blok dijagram Video In to AXI4-Stream [8].*

Iako se iz dijagrama na slici Sl 2.6 može primijetiti da je poželjno uz *Video In* blok koristiti *Video Timing Controller* (VTC) u detekcijskom modu za potrebe ovog projekta to nije imalo utjecaja stoga se VTC nije na ulaznom dijelu ni koristio. Iz razloga što je cilj podatke spremiti u memoriju, a prilikom spremanja sinkronizacijski signali se odbacuju i VTC postaje višak koji oduzima FPGA resurse. Također se na dijagramu može vidjeti način na koji se odvija prilagodba video signala u *AXI4-Sream* podatke, vidi se da se podaci primaju samo ako je video aktivan.

Signal *active_video* detektira se na silazni i uzlazni brid te se spaja na *Asynch* FIFO i dozvoljava mu rad.  Osim piksela primaju se podaci o završetku linije (EOL, engl. *End Of Line*) i početku okvira (SOF, engl. *Start Of Frame*) koji ovise o signalima *active_video* i *vsynch*. EOL je silazni brid *active_video* signala, a SOF je dobiven nakon logičke operacije „I“ na signalima *vsynch* i *active_video*.. *Async* FIFO je unutarnji spremnik varijabilne veličine koji pospješuje pretvorbu i sprječava gubitak podataka zbog kašnjenja ili velikih brzina slanja.

### 2.3.2 *VDMA (Video Direct Memory Access)*

VDMA je blok sa specifičnom primjenom u video aplikacijama. On je sljedeći korak u nizu, nakon što je AXI4-Stream obrađen na željeni način potrebno je spremiti te podatke u memoriju. Zadaća VDMA bloka je upravo to, da što brže i jednostavnije obrađene podatke spremi u fizičku memoriju.




*Sl. 2.7 VDMA jednostavni blok dijagram [9].*

Konfiguracija i kontrola VDMA odvijaju se preko AXI4-Lite protokola preko kojeg je omogućen pristup registrima bloka iz razvojnog okruženja. Nakon što su registri programirani, blok *Control and Status* generira potrebne naredbe za *DataMover* blok koji inicira naredbe za pisanje i čitanje između *AXI4-Stream* i *AXI4 Memory Map* sučelja. Putanje za čitanje i pisanje su neovisne, iako je dozvoljena sinkronizacija ulaznih i izlaznih okvira.

Podržane su različite veličine podatkovnih sabirnica, od 32 bita, do 1024 bita, ali veličina sabirnice mora se poklapati s veličinom podataka koji su na *AXI4-Stream* sučelju [9].

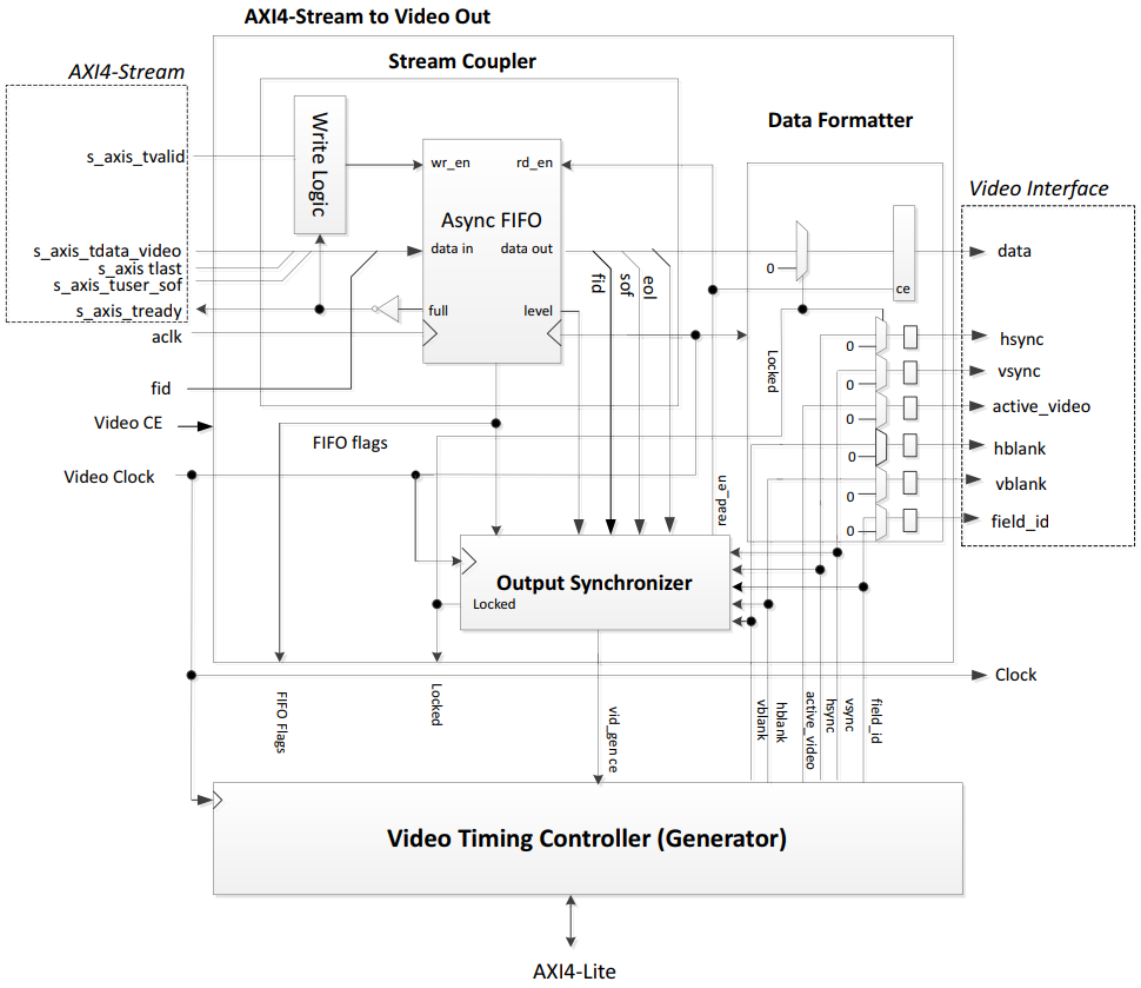
Postoje maksimalno 32 spremnika za okvire (engl. *Frame Buffers*), i mogućnost slanja neporavnate memorije. Slanje neporavnate memorije je opcija koja dozvoljava korisniku zapisivanje proizvoljne količine memorije. Takav način nije ograničen horizontalnom i vertikalnom veličinom okvira, uz to može započeti s bilo koje adrese koja se zada. Ako se koristi opcija *Frame Buffers*, tada je potrebno svakom okviru dodijeliti memoriju koja mu je potrebna.

Sinkronizacijske opcije označavaju ponašanje *master* bloka i slave bloka nad okvirima koji se koriste, te su neophodne ako postoji više VDMA blokova koji su spojeni, jer *master* sinkronizacija (engl. *Genlock Master*) koja je početno zadana, ne uzima u obzir što se događa na još korištenim okvirima, čak i ako se dogodi greška i otpadne okvir, *Genlock Master* nastavlja sa svojim radom normalno [9].

### 2.3.3 *AXI4-Stream to Video Out (Video Out)*

Za potrebe putanje reprodukcije, nužno je formatirati podatke odnosno video koji je spremljen kao niz piksela, u oblik kakav i kamera generira. U tu svrhu koristi se *Video Out* blok. *Video Out* dobiva podatke iz memorije preko VDMA kanala za čitanje. Ovakvi podaci se ne mogu slati jer nemaju definirane sinkronizacijske signale, odnosno takvi podaci ne tvore nikakve okvire. Kako bi se video mogao reproducirati uz same podatke potrebno je generirati sinkronizacijske signale koristeći VTC (*Video Timing Controller*) u modu generatora [10].

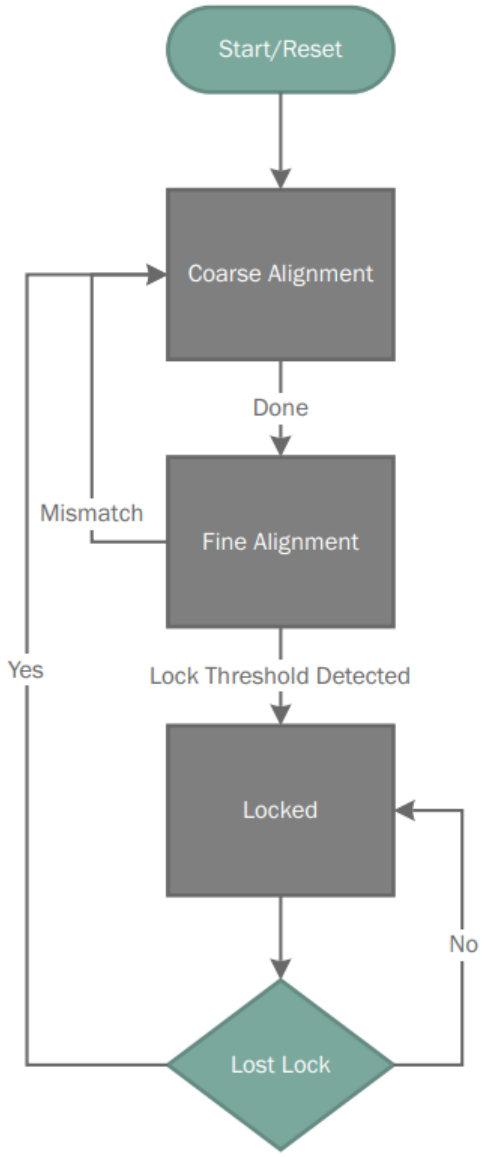
*Video Out* blok konvertira *AXI4-Stream* u izlazni video, koji se sastoji od paralelnih video podataka, horizontalne i vertikalne sinkronizacije, zacrnjivanja, te signala koji označava validnost podataka na sučelju. Ako je potrebno *Video Out* može proširivati ili sužavati piksele, tako što ili nadoda vrijednosti na najniže bitove (LSB) ili obriše vrijednosti s najnižih bitova, odnosno izbaci najniže bitove. Na primjer, ako su pristigli podaci 10 bitni, a izlaz je 12 bitni, *Video Out* proširi ulazne podatke nadodavanjem nula na najniže bitove. [10].




*Sl. 2.8 Blok dijagram za AXI4-Stream to Video Out [10].*

Iz dijagrama sa slike Sl. 2.8 očigledno je kako *Video Out* ne može funkcionirati bez VTC-a. Slično kao i *Video In* ali iz suprotnih razloga potreban je unutarnji spremnik podataka, koji dozvoljava da ulazni podaci dolaze različitom brzinom od one kojom se podaci u obliku videa šalju na izlaz.

Dijagram toka komponente koja sinkronizira izlaz (engl *Output Synchronizer*) prikazan je na slici Sl. 2.9. Ova komponenta ima najbitniju ulogu u bloku *Video Out*, a uloga je sinkronizacija dva ulaza. Jedan ulaz je *AXI4-Stream* preko kojeg dolaze nepravilni i asinkroni video podaci, drugi ulaz je spojen na VTC koji dovodi periodične i pravilne vremenske signale. Podaci sa *AXI4-Stream*-a se spremaju u unutarnji spremnik (*Asynch* FIFO), a VTC-u se dovodi određeno kašnjenje da bi se postiglo poravnanje podataka i vremenskih signala.




*Sl. 2.9 Dijagram toka Output Synchronizer-a [10].*

*Coarse Alignment* je grubo poravnavanje vremenskih signala *i AXI4-Stream* podataka, Detektira se signal SOF(engl. *Start Of Frame*), čeka se na pun spremnik, prelazi se na *Fine Alignment* odnosno precizno poravnavanje [10].

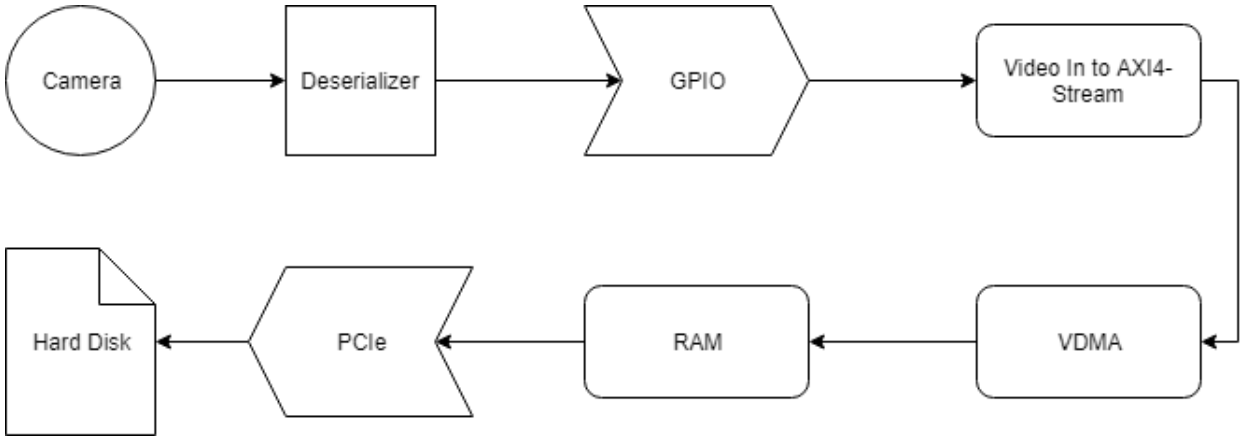
### 2.3.4 *VTC (Video Timing Controller)*

Već spominjani blok, VTC, neophodan za izlaznu putanju, pogodan za ulaznu, je Xilinx®-ovo rješenje za održanje kontinuiranosti video podataka koji za potrebe obrade moraju biti diskretnog tipa. Apsolutno prilagodljiv blok, uz ponuđene često korištene video formate dozvoljava korisniku konfiguraciju vlastitih video formata. Koristeći *AXI4-Lite* protokol, konfiguracija je moguća i preko programskog okruženja, samim time i iz aplikacijskog sloja [11].

VTC podržava dva načina rada, detektor način za očitavanje sinkronizacijskih signala na ulaznom videu i generator način za stvaranje sinkronizacijskih signala. Dobar primjer korištenja obje funkcionalnosti bi bio slučaj u kojem se FPGA koristi za Gaussovo filtriranje okvira i konačno prikaz takvog videa. Na ulazu je potrebno detektirati sinkronizacijske signale koristeći VTC u detektor modu, a podatke slike izdvojiti i formatirati u *AXI4-Stream* za video obradu koja vrši gaussovo filtriranje, te koristeći *Video Out* s VTC generiranim signalima koji su identični detektiranima, prikazati taj video na izlazu [11].

# 3 Razrada problema

Putanja snimanja prikazana je slici Sl. 3.1 Dijagram obuhvaća i dijelove izvan FPGA dizajna, također i PCIe sučelje kojim se ne bavi ovaj rad.

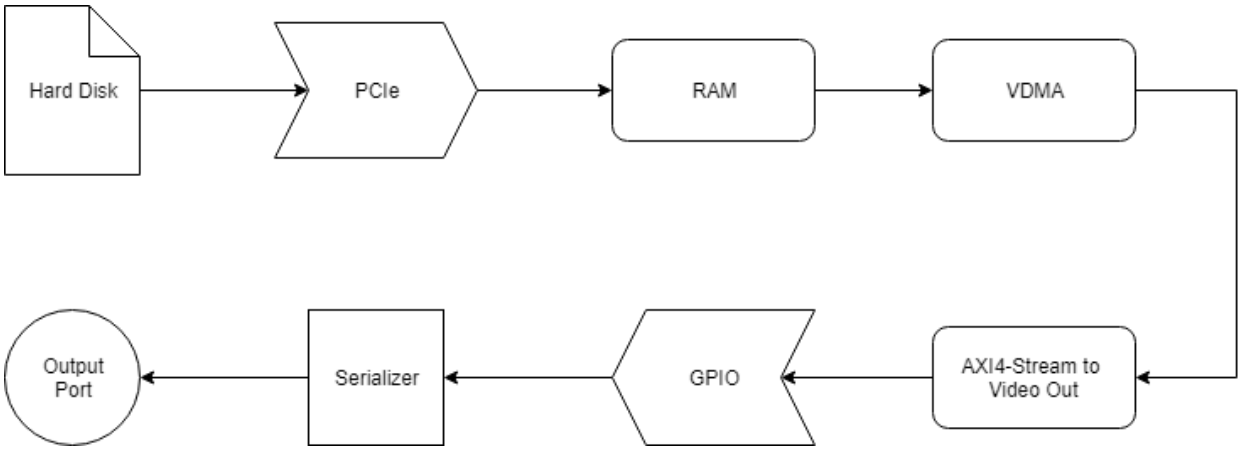



*Sl. 3.1 Dijagram toka putanje snimanja.*

Podaci dolaze od kamere putem koaksijalnog FPD-Link III do *deserializer*-a. Prolaskom kroz *deserializer* podaci su preoblikovani u paralelni format te prihvaćeni na višenamjenske ulazno-izlazne pinove FPGA-a. Sljedeći dio putanje *Video In*-VDMA-RAM je dizajn ulazne putanje kojim se ovaj radi bavi. Odnosno dizajnirati FPGA tako da primi video podatke, pretvori ih u *AXI4-Stream*, spremi ih u memoriju i prepusti ih dalje PCIe upravljaču. Konačna destinacija snimljenih podataka je tvrdi disk.

Funkcija ove putanje je da snimljeni podaci pristignu u paketima do računala koje upravlja pločom preko aplikacije. Željeni rezultat je sposobnost snimanja na svih 9 portova, korištenjem ili jednog ili drugog tipa kamere, i istovremeno snimanje do 9 kamera.

Putanja reprodukcije ili izlazna putanja je obrnuta putanja koja koristi velik broj istih blokova kao i ulazna putanja i prikazana je na slici Sl. 3.2.




*Sl. 3.2 Dijagram toka putanje reprodukcije*

Za ovaj slučaj izvor podataka je tvrdi disk, koji prijašnje snimljene video podatke preko PCIe sučelja dovodi do radne memorije ploče. PCIe upravljač signalizira VDMA upravljaču da je spremanje u memoriju završeno, VDMA se uključuje i čita podatke iz memorije koje *Video Out* blok uz pomoć VTC-a oblikuje u video format i dovodi do GPIO sučelja. Izvan FPGA dijela treba pokrenuti *serializer* da paralelne podatke pretvori u serijske i odvede do izlazni portova AMV Grabber uređaja.

Slanje podataka daje ovoj ploči funkcionalnost emuliranja kamere. Ishod koji se želi postići je da se AMV Grabber ploča nadoveže na drugu ADAS ploču sa svrhom testiranja raznolikih ADAS algoritama u laboratoriju bez potrebe za vožnjom prilikom svakog testiranja, također pruža jednolične materijale za testiranja za različite ploče. Reprodukcija nije ograničena brojem izlaznih portova, ali je ograničava dijeljenje podatkovnih sabirnica GPIO pinova programibilne logike. Odnosno, iste GPIO pinove koriste ulazni i izlazni portovi, samim time ako je *serializer* spojen u svrhu slanja podataka tada te pinove *deserializer* ne može koristiti.

Ponuđeno rješenje zamišlja primjenu ploče u dva slučaja s dvije različite konfiguracije, kada ploča snima, koristi ulazne portove niti jedan izlaz. Prilikom snimanja raspored korištenih kamera je proizvoljan s obzirom na zahtjeve korisnika, npr. prvih pet portova za kamere tipa 1) i ostala četiri porta za kamere tipa 2). Prilikom slanja podataka svi GPIO pinovi su oslobođeni za emuliranje kamera, tako da se niti jedan port ne može koristiti za snimanje.

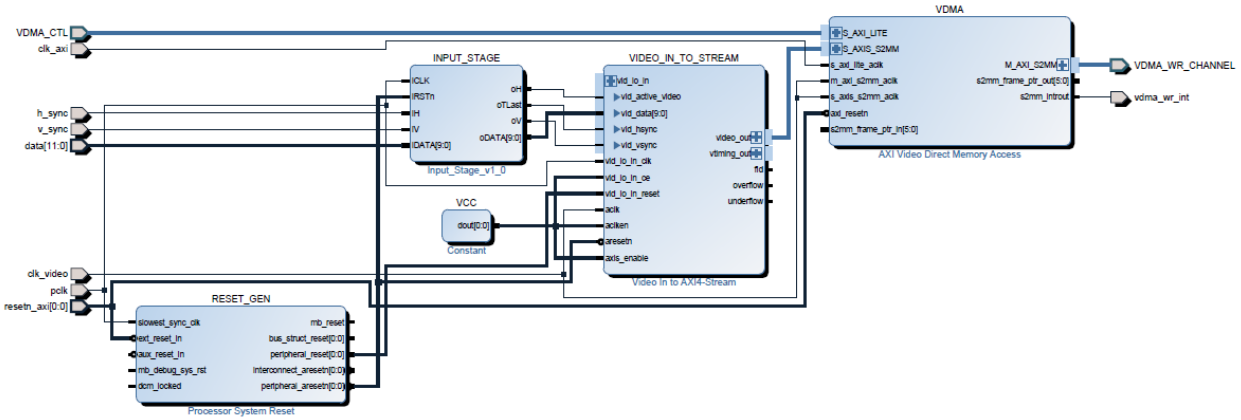
# 4 Rješenje problema

Rješenja su razdvojena, jer u konačni proizvod nije bilo moguće implementirati putanju reprodukcije zbog poteškoća s prijenosom podataka preko PCIe sabirnice na memoriju ploče. Prvi dio putanje reprodukcije od računala do memorije ploče nije funkcionirao jer se ispostavilo da pri pokušajima zapisivanja na ploču poremete adrese gdje treba zapisati. Grešku nije moguće bilo na vrijeme ispraviti jer uzrok bio pogrešno dizajniran IP blok u korištenoj verziji *Vivado*® okruženja za dizajniranje. Stoga je u ovom poglavlju osvrt na dva odvojena rješenja što se tiče putanje, te potrebne nadopune zbog razlike u kamerama.

## 4.1 Ulazna putanja

Ulazna putanja je putanja snimanja koja je opisana detaljnije u prethodnom poglavlju. Dio te putanje koji je izveden u svrhe ovog diplomskog rada jest od GPIO pinova do PCIe sabirnice.

Kako bi se što bolje objasnilo rješenje putanje snimanja priložena je slika najvažnije hijerarhije blokova u FPGA dizajnu putanje snimanja.




*Sl. 4.1 Video Input blok dizajn*

Korišteni blokovi su:

- *Video In to AXI4-Stream* – već opisani blok u vlastitom potpoglavlju *Video In to AXI4-Stream*, služi za formatiranje video podataka u *AXI4-Stream*.
- VDMA – Direktno spojena na *Video In to AXI4-Stream* s ulazne strane, s izlazne je spojena preko *AXI Interconnect* bloka na *Zynq7 Processing System* preko kojeg odlaze podaci do memorije.

- *Input Stage* – pomoćni blok koji služi kao *debouncer* primljenih podataka, odnosno radi zadršku od dva takta za svaki primljeni piksel. Ovakvim pristupom osigurava se postojanje podataka na ulazu u *Video In to AXI4-Stream*, i smanjuje se utjecaj gubitaka.
- *Processor System Reset* – služi za sinkronizaciju svih procesa koji se koriste u dizajnu
- *Constant* – vrlo jednostavan blok koji predstavlja napajanje.

Pristigli sinkronizacijski signali mogu biti direktno spojeni na *Video In*, nakon čega se više nigdje ne koriste. Iz tog razloga VTC blok nije potreban, a pristigli podaci na VDMA spremaju se u memoriju bez sinkronizacijskih signala.

Konfiguracija *Video In* bloka mora biti napravljena u samome dizajnu. *Video In* ne podržava *AXI4-Lite* protokol zbog čega mu je nemoguće pristupiti preko razvojnog okruženja. Parametri konfiguracije za slučaj korištenja kamere tipa 1) su sljedeći:

- Širina podataka na ulazu je 12 bitova.
- Širina podataka *AXI4-Streama* na izlazu je 16 bitova (mora biti jednaka VDMA kanalu za pisanje)
- Prenosi se 1 piksel po taktu
- Format videa je RAW odnosno opcija u bloku se zove *Mono/Sensor*
- Dubina spremnika je 2048
- Takt je neovisan, jer kamera šalje svoj piksel takt koji se vrlo vjerojatno neće poklapati s taktom koji upravlja sve komponente.

Jedina razlika u konfiguraciji što se tiče kamera tipa 1) i 2) je u širini video podataka na ulazu, u slučaju kamere 2) širina je 10 bitova [4] [5]. Iako je drugačiji format koji kamera šalje, YUV, format videa pod nazivom *Mono/Sensor* govori da blok propušta piksel po piksel, neovisno je li prvi piksel „Y“ ili „V“ odnosno „R“ ili „G“, korisnik mora biti svjestan kakav raspored piksela kamera šalje [8]. U tom modu *Video In* propušta piksele kako su i došli, naravno formatirane na *AXI4-Stream*.

VDMA je konfigurirana za pisanje, kanal za čitanje je potpuno isključen. Iako VDMA podržava *AXI4-Lite* sučelje, ukoliko u blok dizajnu ne postoji kanal kojeg se pokušava konfigurirati, doći će do greške. Osim same generacije kanala VDMA bloka i parametara tih kanala, sva konfiguracija vrši se u razvojnom okruženju, najvažnije stavke su:

- Visina, odnosno broj linija u okviru - ovisno o kameri: 1) 1080, 2) 720.
- Širina, broj piksela u liniji – ovisno o kameri, za ovaj slučaj je jednako 1280*2, množenje s dva jer spremljeni video format ima 2 byte-a po piskelu.
- Stride, aktivna veličina linije – također 1280*2
- Dodjela adrese, koristi se 10 *Frame Buffer*-a svakom od njih je potrebna adresa koja pokazuje na zadanu veličinu okvira.

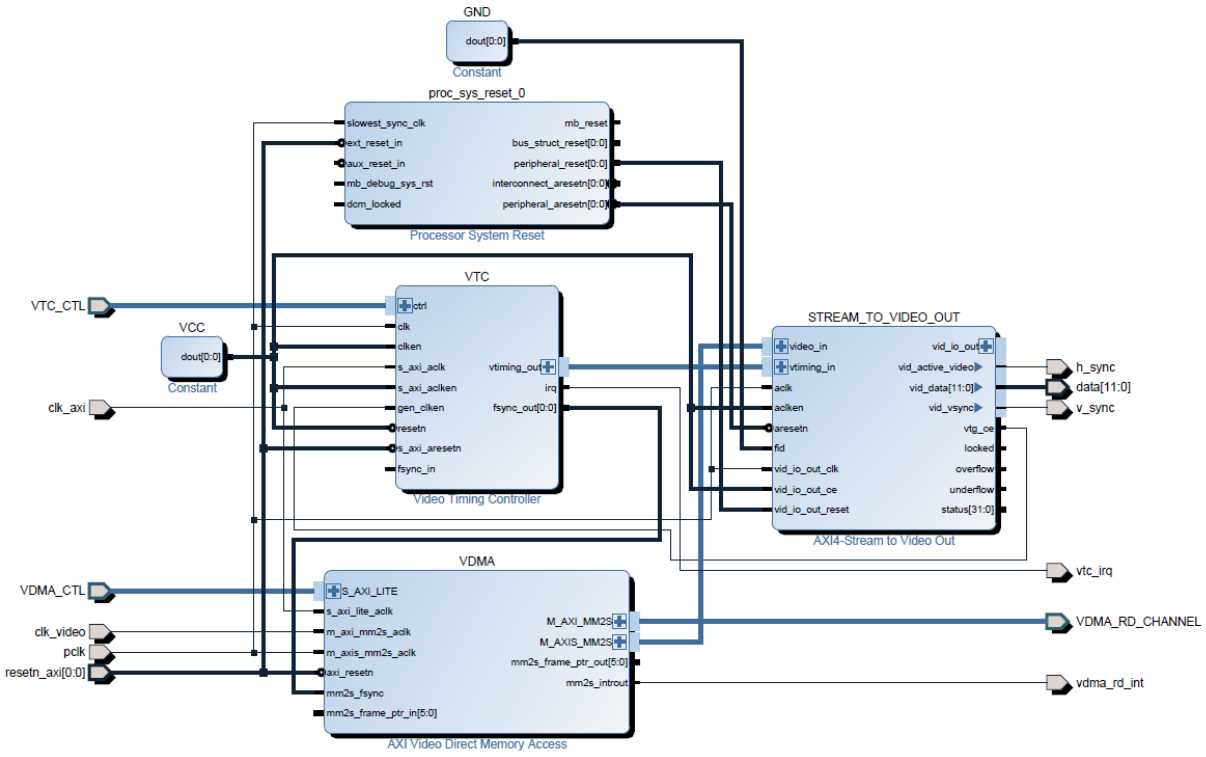
Postupak konfiguracije odvija se od inicijalizacije instance VDMA-e, preko uspostave kanala pisanja, do dijela gdje se zadaje povratna funkcija (engl. *Callback*), na bilo koji prekid koji je dozvoljen u konfiguraciji. Postoje dvije *Callback* funkcije, jedna za uspješan završetak posla, druga u slučaju greške (engl. *ErrorCallback*).

Protok podataka odvija se tako da nakon svakih deset okvira koje VDMA uspješno primi dogodi se prekid (engl. *Interrupt*), poziva se *Callback* funkcija koja provjerava jesu li sve aktivne VDMA-e primile po deset okvira, ako jesu, slijedi dodjela novih deset adresa za svaki *Frame Buffer* za svaku VDMA instancu. Prethodno korištena memorija oslobađa se za PCIe prijenos podataka, a VDMA-e se pokreću s novim adresama spremne za novih deset okvira. Takva izmjena adresa *Frame Buffer*-a za svakih deset okvira traje dok god je snimanje aktivno.

Ovako dizajniran i konfiguriran blok uz podršku na razvojnom okruženju uspješno odrađuje svoju zadaću. Konačni dizajn je jednak samo je umnožen broj *Video Input* blokova devet puta. Omogućava istovremeno snimanje do pet kamera, iz razloga što na hardverskoj platformi nisu ni svi portovi funkcionirali, i zbog podatkovnog uskog grla na tvrdom disku koje ne daje veću propusnost od pet kamera.

## 4.2 Izlazna putanja

Zbog poteškoća u prenošenju podataka s računala na ploču, implementacija izlazne putanje u konačan proizvod nije bila moguća. Stoga je dizajn sastavljen od samo jednog izlaznog bloka koji odrađuje reprodukciju podataka s memorije ploče. Memorija je inicijalizirana na uobičajeni okvir testnog obrasca (engl. *Test Pattern Frame*) za slanje i provjeru valjanosti. Ovako prilagođen dizajn izlazne putanje mogao se odvojeno razvijati i testirati.




*Sl. 4.2 Video Output blok dizajn*

Korišteni blokovi su:

- *AXI4-Stream to Video Out* – blok koji formira izlazne video signale iz ulaznog *AXI4-Stream*-a i generiranih sinkronizacijskih signala iz VTC-a.
- VDMA – Direktno spojena na *Video Out* s izlazne strane, s ulazne je spojena preko *AXI Interconnect* bloka na *Zyng7 Processing System* preko kojeg dolaze podaci od memorije.
- VTC – u generator modu rada, osigurava postojanje sinkronizacijskih signala
- *Processor System Reset* – služi za sinkronizaciju svih procesa koji se koriste u dizajnu
- *Constant* – u ovom dizajnu postoje dva, jedan je uzemljenje drugi je napajanje.

VTC podržava *AXI4-Lite* sučelje, poželjno je u samome dizajnu postaviti početnu konfiguraciju, iako se treba u razvojnom okruženju odviti konfiguracija ovog bloka. U VTC-u se definira veličina aktivnih piksela; kamera 1) 1280x1080 ili kamera 2) 1280x720, veličina linije i okvira uz signale zacrnjivanja koja teoretski može biti bilo kakva jer se ne uzima u obzir prilikom slanja.

*Video Out* je konfiguriran s obzirom koju kameru emulira:

- Širina podataka na izlazu je 12 bitova, ili 10 bitova ako je kamera 2).
- Širina podataka AXI4-Streama na ulazu je 16 bitova (mora biti jednaka VDMA kanalu za čitanje)
- Prenosi se 1 piksel po taktu
- Format videa je RAW odnosno opcija u bloku se zove *Mono/Sensor*
- Dubina spremnika je 32
- Takt je neovisan, jer se takt ulaznih podataka *AXI4-Streama* ne poklapa s taktom signala koji su generirani u VTC-u.
- Histereza je razine 12

VDMA je konfigurirana za čitanje, dok je kanal za pisanje potpuno isključen. Iako VDMA podržava *AXI4-Lite sučelje*, ako u blok dizajnu ne postoji kanal kojeg se pokušava konfigurirati, doći će do greške. Osim same generacije kanala VDMA bloka i parametara tih kanala, sva konfiguracija se vrši u razvojnom okruženju, najvažnije stavke su:

- Visina, odnosno broj linija u okviru - ovisno o kameri: 1) 1080, 2) 720.
- Širina, broj piksela u liniji – ovisno o kameri, za ovaj slučaj je jednako 1280*2, množenje s dva jer spremljeni video format ima 2 byte-a po piskelu.
- Stride, aktivna veličina linije – također 1280*2
- Dodjela adrese, koristi se 10 *Frame Buffer*-a svakom od njih je potrebna adresa koja pokazuje na zadanu veličinu okvira.

Postupak konfiguracije je sličan kao i za putanju snimanja, razlika je u tome što se u ovom slučaju povratne funkcije (engl. *Callback)* pokreću kada je čitanje iz memorije završeno ili neuspješno. Iz razloga što ova putanja nije implementirana u konačni proizvod samo jedna instanca VDMA-e je korištena. Stoga je protok podataka bio sljedeći: nakon što se pokrene reprodukcija, pokreće se i rad VDMA-e, koja krene iščitavati zadanu memoriju u kojoj je spremljen *Test Pattern* okvir, nakon što je pročitano deset okvira, događa se prekid (engl. *Interrupt)* na kanalu za čitanje, promjene se adrese za *Frame Buffer*-e, do sada korištena memorija se oslobađa za PCIe koji bi trebao upisati nove podatke, a VDMA se pokreće s novim adresama.

Rezultat ovako dizajnirane putanje za reprodukciju nije ispunio sve zahtjeve, iz nekog razloga VTC i VDMA nisu pravovremeno postizali sinkronizaciju, nego im je bilo potrebno pet okvira koja VDMA uspješno pročita. Tek nakon tih pet okvira sinkronizacija je postignuta uključuje se izlaz *Video Out* bloka i slijedeći okviri se uspješno šalju na izlaz.

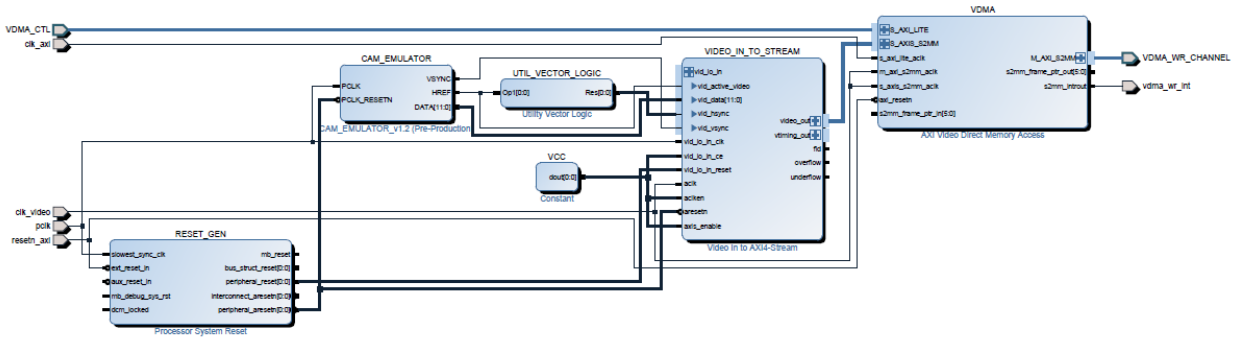
Revizijom literature [10] saznalo se da u *Video Out* bloku *Output Synchronizer* komponenta zahtjeva 4 okvira za uspostavu *lock* signala. Preostalo je pitanje kako zaustaviti VDMA čitanje dok se ne dogodi sinkronizacija, odnosno ponavljanje prvog okvira do sinkronizacije pa tek onda prelazak na druge okvire. U ovom radu je predloženo rješenje: nekorištenje memorije prvih pet okvira za informacijama koji trebaju doći preko PCIe sabirnice nego se popune sa „sinkronizacijskim“ podacima koji će biti potpuno crni okviri. Također, povećanje broja *Frame Buffer*-a na 15, što bi dovelo do željene funkcionalnosti.

# 5 Rezultati testiranja

Testiranje FPGA blokova ima svoje posebne metode. Jedna metoda je korištenje simulacija koje *Vivado*® omogućava, odnosno pregled ponašanja signala ako im se virtualno zadaju ulazni parametri. Simulacija pomaže u verifikaciji korektnog ponašanja dizajna. Potrebno je napraviti *test bench*, specificirati opcije simulacije i povezati s potrebnim bibliotekama radi pravilnog rada sustava. Ako se simulacija radi nakon sinteze ili implementacije potrebno je i generirati *Netlist*. Preostaje samo pokrenuti simulator, koji može biti integriran u *Vivado*® ili simulatori trećih stranaka [12].

Druga metoda testiranja FPGA dizajna, je nadzor signala u stvarnom vremenu korištenjem blokova *Integrated Logic Analyzer* (ILA). Široko primjenjiv i prilagodljiv blok, sinkron je s taktovima koji se koriste u FPGA dizajnu, omogućava raznolik broj signala koji se nadziru i količinu podataka koja će biti uhvaćena za pojedini signal. Signali dizajna se paralelno spajaju s ILA sondama, te se s obzirom na takt spremaju podaci signala u BRAM (engl. *Block RAM*) programibilne logike. ILA ima direktnu povezanost s JTAG sučeljem *SoC*-a, koja omogućava prikaz valnih oblika u *Vivado*®-u [13].

## 5.1 Testiranje ulazne putanje




*Sl. 5.1 Dizajn ulaznog bloka za potrebe testiranja.*

Testiranje ulazne putanje, odnosno dijela ulazne putanje od *Video In* bloka do tvrdog diska izvršeno je koristeći modificirani dizajn ulaznog bloka, prikazan na slici Sl. 5.1.. Funkcionalnost portova, odnosno činjenica da ni jedna AMV Grabber ploča nije imala funkcionalnih svih 9 portova, stvorila je potrebu za emuliranjem kamera na ulazu. Dizajn je gotovo isti kao i za slučaj s kamerama, razlika je što su ulazni pinovi zamijenjeni CAM_EMULATOR blokom vlastite

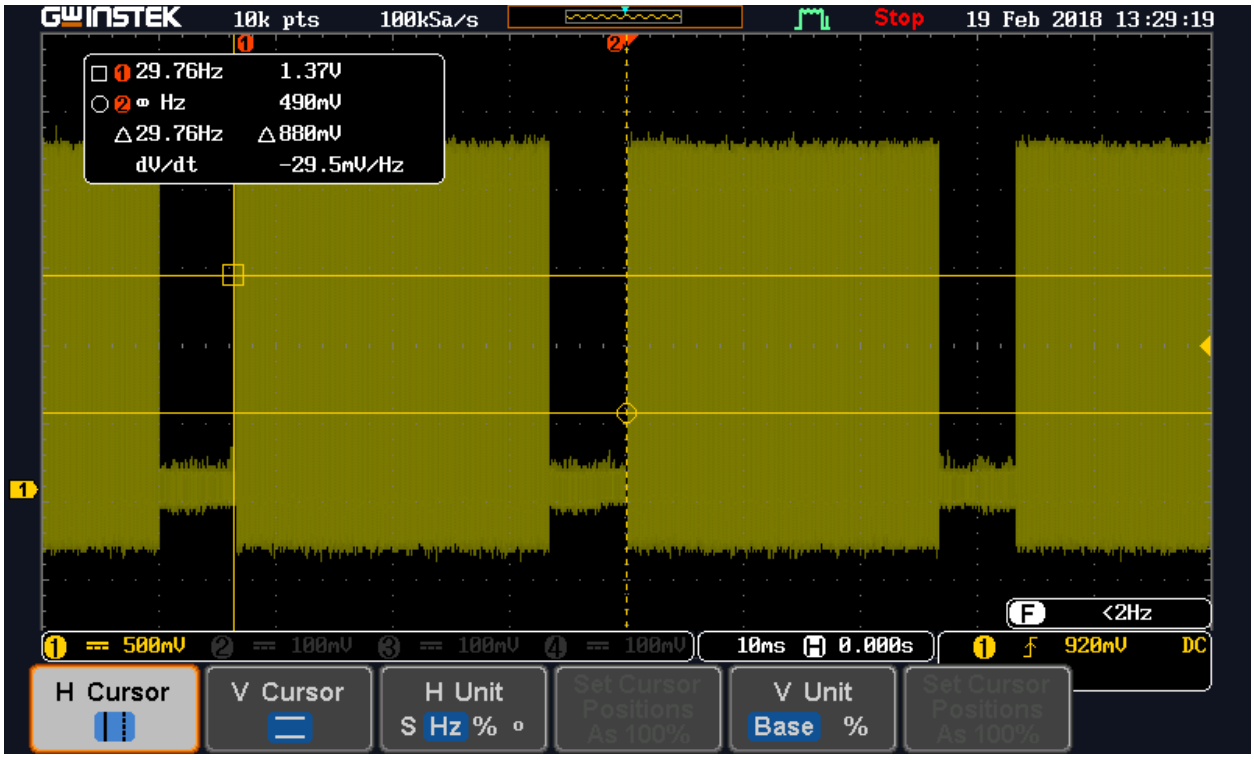
izrade koji generira *Test Pattern* okvir s namjerom da se ispita cjelokupna putanja opterećena s 9 kamera.

Rezultat ovog ispitivanja je pokazao da je „usko grlo“ tvrdi disk, a FPGA i PCIe su sposobni potrebnu količinu podataka prenijeti. Računajući po rezoluciji i FPS-u kamere tipa 1):

$$(2 * 1280 * 1080)byte * 30\frac{1}{s} = 79\frac{MB}{s},$$

Devet kamera bi stvaralo protok podataka od 711 $MB/s$. Test je napravljen na SSD disku, te je uspješno snimljeno koristeći maksimalno pet emulatora. PCIe sabirnica na ploči ima četiri kanala stoga je njen teorijski maksimalni protok 2 $GB/s$, a FPGA ima svoj maksimum na 1 $GB/s$.

Činjenica da nisu svi ulazni portovi bili funkcionalni je provjerena nadgledanjem sinkronizacijskih signala na *deserializer*-u korištenjem osciloskopa. Valjani portovi su pokazivali valne oblike horizontalne i vertikalne sinkronizacije. Iz valnih oblika bilo je vidljivo da kamera šalje jedan okvir svakih ≈32ms što je ekvivalentno 30 FPS. Portovi koji nisu radili imali su šum za valni oblik.

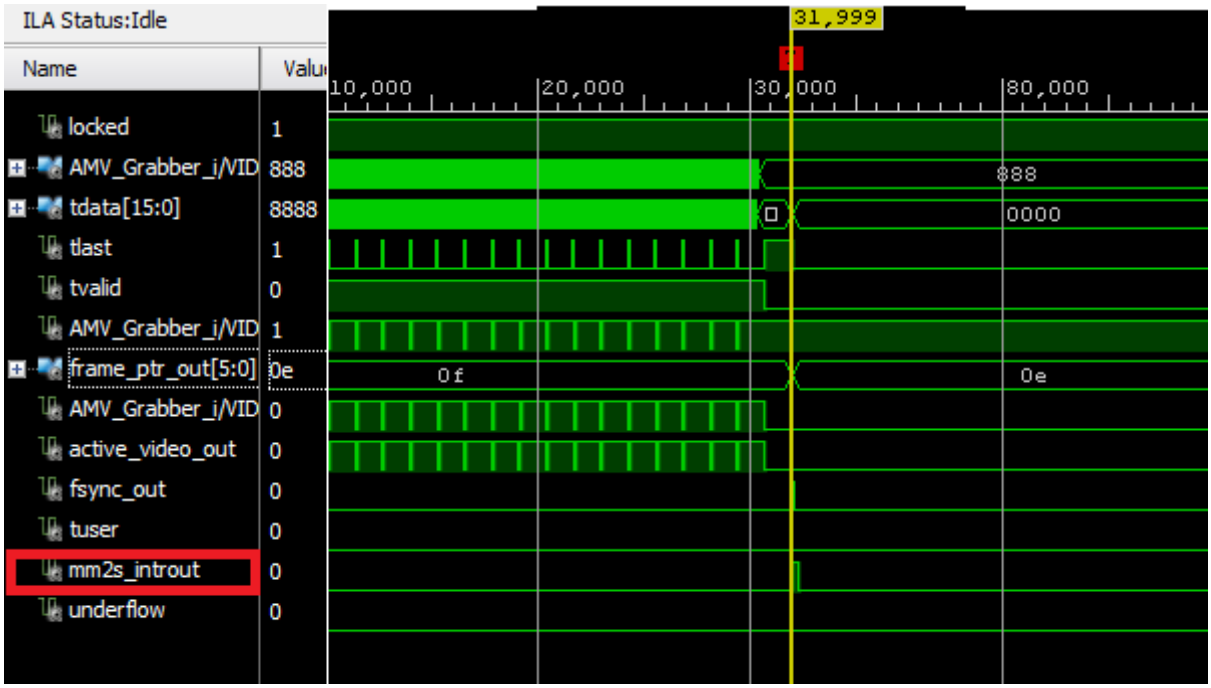



*Sl. 5.2 Horizontalna sinkronizacija na deserializeru uhvaćena osciloskopom.*

Slika (Sl. 5.2) prikazuje signal horizontalne sinkronizacije na *deserializer*-u valjanog porta. Markeri označavaju početke dva okvira, a njihova razlika po x-osi je 29.76Hz odnosno 30FPS. Između dva markera nalazi se 1080 signala horizontalne sinkronizacije i signal zacrnjivanja.

## 5.2 Testiranje izlazne putanje

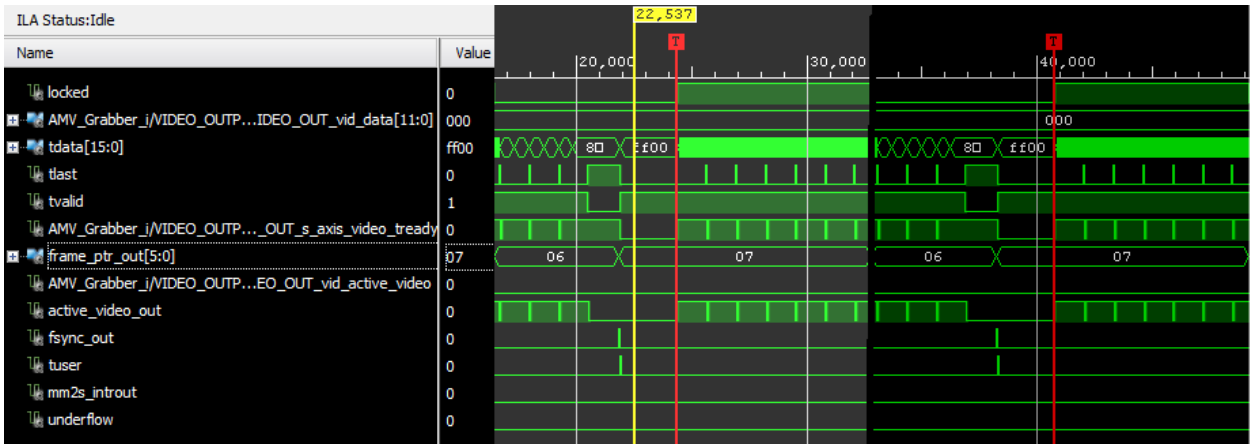
Korištena je metoda testiranja ILA blokovima, i kao konačna potvrda testiranja su vršena spajanjem dvije AMV Grabber ploče, jedna šalje *Test Pattern* video, a druga prima. Na prvi pogled je izgledalo da je primljeni video jednak poslanom, jer je video napravljen ponavljanjem istog *Test Patte*rn okvira, ali podaci uhvaćeni ILA blokom sugerirali su da nešto nije dobro.




*Sl. 5.3 ILA podaci uhvaćeni pod uvjetom da se signal mm2s_introut dogodio.*

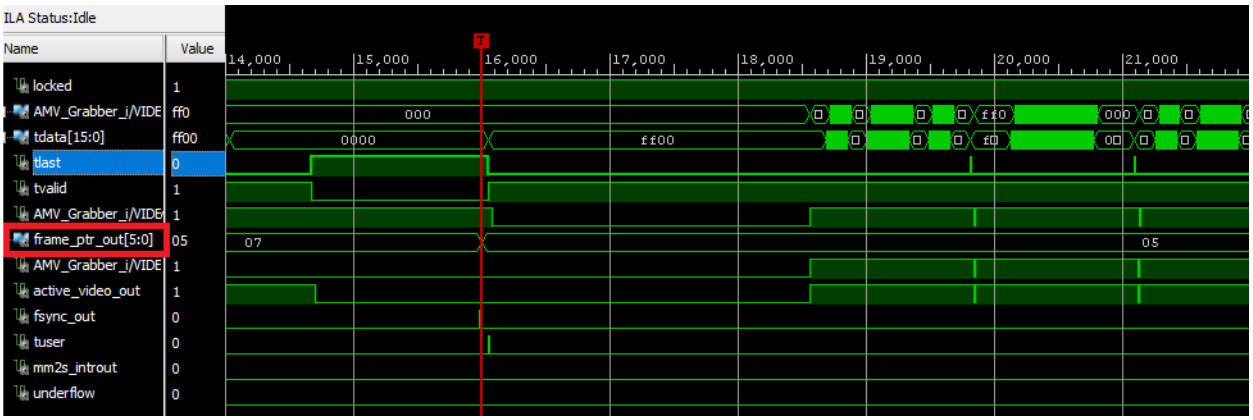
Slika Sl. 5.3 pokazuje da je VDMA uspješno pročitala 10 okvira. Signal *mms2s_introut* je *Interrupt* VDMA-e, u ovom slučaju jedinična pojava tog signala na promjeni okvira pokazuje uspješno čitanje (broj uzoraka na slici 31999). U slučaju *Interrupt*-a na grešku taj signal bi imao u kratkom vremenu više pojavljivanja. Promjena okvira se vidi u signalu *frame_ptr_out* koji koristi Gray-ev kod za zapisivanje indeksa okvira, a ta promjena je iz $0F_{(16)}$ u $0E_{(16)}$. Nakon te promjene vidimo da nema podataka jer je prekid zaustavio rad VDMA-e te se čeka na pripremu nove memorije i ponovnim pokretanjem, kao što je opisano u prethodnom poglavlju način na koji VDMA čitanje funkcionira.

Ovdje se već pronalazi jedan problem, jer Gray-ev kod nalaže da su $0F_{(16)}$ i $0E_{(16)}$ uistinu deveti i deseti okvir. Daljnja testiranja su potrebna za otkrivanje problema. Sljedeće dvije slike prikazuju gdje prijenos podataka počinje.




*Sl. 5.4 Uvjet aktivacije ILA-e je signal LOCKED.*

Slika Sl. 5.4 prikazuje dva odvojena slučaja kada je aktiviran *lock* signal. Svjetliji dio je za prvih 10 okvira, a tamniji dio se odnosi na drugih 10 okvira. U oba slučaja u isti trenutak se aktivira *lock*.




*Sl. 5.5 Okidač je ovjde zamaknut tako da signal LOCKED bude visoko, i da se dogodila promjena frame_ptr_out signala*

Signal *locked* ukazuje na to da su VDMA i VTC sinkronizirani te da su podaci na *Video Out* bloku validni i spremni za slanje. Tek kada je *locked* visoko, slanje može započeti. Problem je što se *locked* postavio visoko tek na $07_{(16)}$ okviru što je po Gray-evom kodu 4, a slanje podataka nije počelo do sljedećeg okvira $05_{(16)}$, peti okvir. Drugi problem je što je VDMA bez obzira na nedostatak sinkronizacije propustila tih četiri okvira. Kako to izgleda za svih deset okvira pokazuje sljedeća slika.


*Sl. 5.6 Deset ILA okidača na promjenu okvira, slika donja slika je produljenje gornje.*

Signal *tuser* šalje VDMA kada je pročitala jedan okvir i taj je signal sinkron *fsync_out* signalu kojeg proizvodi VTC nakon što generacija sinkronizacijskih signala završi za jedan okvir.

Predloženo rješenje koje minimizira gubitke okvira i održava brzinu slanja, nalaže da se količina *Frame Buffer*-a koje VDMA koristi poveća na 15, a prvih pet okvira koji zauzimaju memoriju popune crnim pikselima, dozvoljavajući tako potrebno vrijeme sinkronizacije VTC-a i VDMA-e, također i slanje deset okvira koje je zamišljeno. Takav rad prikazuje slika Sl. 5.7


*Sl. 5.7 Predloženo rješenje s 15 korištenih Frame Buffer-a pokazuje da se željenih deset okvira može poslati, uz cijenu pet sinkronizacijskih okvira.*

Na slici, poslani okviri se očitiju aktivacijom *vid_active_video* i *vid_data* signala uz *locked* postavljen visoko, od $05_{(16)}$ do $08_{(16)}$ vrijednosti *frame_ptr_out*.

# Zaključak

Zahtjevi ovog zadatka su gotovo potpuno ispunjeni. Rješenja su razdijeljena u dva dijela, ulazna putanja koja je putanja snimanja podataka s kamere i izlazna putanja koja reproducira ranije snimljen sadržaj na izlazne portove.

Ulazna putanja omogućava maksimalni protok video podataka koje hardverska ograničenja dozvoljavaju. Usko grlo koje se pojavljuje pri zapisivanju podataka na tvrdi disk, može se riješiti korištenjem bržih SSD diskova.

Izlazna putanja nije implementirana u konačno rješenje AMV *Grabber* projekta, ali testiranjem je pokazano da može poslati željen broj okvira, ako se kašnjenje zbog sinkronizacije nadomjesti s dodatnim zauzećem memorije. Odnosno, moguće je poslati proizvoljan broj okvira jednim uključivanjem VDMA-e dok god se osigura pet okvira za sinkroniziranje. Jedino je preostao problem kako narediti da VDMA ponavlja prvi okvir dok se ne postigne sinkronizacija.

# Literatura


[1] P. J. Ashenden, *The Designer's Guide to VHDL.pdf*, 3rd ed. Systems on Silicon, 2008.
[2] M. Krbanjević, I. Rešetar, V. Škobić, "Univerzalna platforma za ispitivanje uređaja za mašinsku vizuelnu percepciju okoline u sistemima za pomoć u vožnji i autonomno kretanje vozila," *Zbornik 61. Konferencije za elektroniku, telekomunikacije, računarstvo, automatiku i nuklearnu tehniku, ETRAN 2017*, p. 3, 2017.
[3] "Zynq-7000 All Programmable SoC Technical Reference Manual (UG585)," p. 1863, 2016.
[4] "OV10640, datasheet." OmniVision Technologies, 2015.
[5] "OV10635, datasheet." OmniVision Technologies, 2015.
[6] "DS90UB91xA-Q1 25-MHz to100-MHz 10/12-Bit FPD-Link III Serializer and Deserializer," *Texas Instruments*, p. 72, 2014.
[7] "AXI Reference Guide," vol. 13, p. 82, 2011.
[8] "Video In to AXI4-Stream v4.0 LogiCORE IP Product Guide (PG043)," p. 43, 2017.
[9] "AXI Video Direct Memory Access v6.2 LogiCORE IP Product Guide (PG020)," p. 89, 2016.
[10] "AXI4-Stream to Video Out v4.0 LogiCORE IP Product Guide (PG044)," p. 54, 2017.
[11] "Video Timing Controller v6.1 LogiCORE IP Product Guide (PG016)," p. 90, 2017.
[12] "Vivado Design Suite User Guide: Logic Simulation (UG900)," p. 184, 2015.
[13] "Integrated Logic Analyzer v6.1 LogiCORE IP Product Guide (PG172)," p. 31, 2016.

# Sažetak

Zadatak ovog diplomskog rada je napraviti FPGA dizajn koji omogućava AMV *Grabber* ploči primanje i zapisivanje podataka s kamera, te slanje već snimljenog video sadržaja na izlazne portove. AMV *Grabber* ploča ima ulogu produžetka ADAS (engl. *Advanced Driver Assistance System*) uređaja, jer AMV *Grabber* omogućava lakši razvoj i testiranje ADAS algoritama.

Dizajniranje je odrađeno u *Xilinx*®-ovom okruženju *Vivado*®, te su korišteni službeni IP blokovi koje regulira *Xilinx*®. Rješenje je razdvojeno na dva dijela, putanja snimanja i putanja reprodukcije iz razloga što nije bilo moguće putanju reprodukcije implementirati u kompletan projekt. Omogućeno je snimanje do devet kamera, iako hardverska ograničenja nisu dozvoljavala dovoljan protok podataka, usko grlo se pojavilo pri zapisivanju podataka na tvrdi disk. Putanja reprodukcije je razvijena zasebno i potvrda njene funkcionalnosti zahtijevala je pozamašna testiranja. Ispostavilo se da je slanje moguće, uz mali gubitak pet početnih okvira koje je nemoguće sačuvati jer su potrebni za sinkronizaciju blokova u dizajnu reprodukcije.



**Ključne riječi**: AMV Grabber, FPGA, ADAS, kamera, snimanje, reprodukcija, Vivado

# Abstract

The purpose of this master thesis is to create an FPGA design, which allows capturing data from cameras, and reproduction of already captured data to the output ports on AMV Grabber device. AMV Grabber device acts as an extension of ADAS (Advanced Driver Assistance Systems) which allows easier development and testing of ADAS algorithms.

The design has been made in Vivavo® design suite by Xilinx®. For the purposes of the design official IPs have been used. Developed design is separated into two parts, input path, and output path, because output path was not possible to implement into the complete project. Input path is able to record with up to nine cameras, even though only theoretically. Actual capabilities were the recording with five cameras because there happened to be a data flow bottle neck when writing on the SSD drive, and no AMV Grabber device had all nine input ports functional. Output path was developed on its own, so the verification of requested behavior was extensive. The final result was successful reproduction of test data with first five frame loss, due to synchronization process of the IPs used in the output design.



**Keywords**: AMV Grabber, FPGA, ADAS, camera, record video, play video, Vivado

# ŽIVOTOPIS

Ivan Vido je rođen 27.10.1992. u Požeškoj bolnici u Hrvatskoj. Pohađao je Osnovnu školu Dobriše Cesarića nakon čega nastavlja obrazovanje u Prirodoslovno matematičkom odijelu Požeške gimnazije. Za vrijeme obaveznog školovanja stekao je znanja iz matematike, fizike i informatike što ga je odvelo u smjeru tehničkih fakulteta. Položenom državnom maturom upisuje Fakultet elektrotehnike i računarstva u Zagrebu, ali nakon dvije godine zbog različitih poteškoća prebacuje se na srodni fakultet u Osijeku, tadašnji Elektrotehnički fakultet. Njegova trenutna akademska titula je sveučilišni prvostupnik stečena 2016. godine. Na kraju treba spomenuti da je stipendist u RT-RK Institutu, odjela u Osijeku.